テーマ１：指数関数時間アルゴリズム（その２）

本日の講義の目的

- 点彩色問題の指数時間アルゴリズム
- 動的計画法（この場合は表利用計算法とでも言うべき）

1.3. 点彩色問題の指数時間アルゴリズム

重要な NP-型最適化問題の一つである点彩色問題（以下が定義）に対して単純な場合より大幅に効率のよい指数時間アルゴリズムを設計してみよう.

点彩色問題 (COLOR: vertex COLORing problem)
入力： 無向グラフ $G = (V, E)$.
仕事： G の最小点彩色法，あるいは最小彩色数を求めよ.

以下では説明を簡単にするために最小彩色数を求める問題を考える．ただし，以下のアルゴリズムを彩色法を求めるように変更することは容易だろう.

まず，単純な方法を考えてみる．要するに，$K = 1, 2, 3, ..., n$ に対して，K 色で塗り分けられるかを調べればよい．それには頂点集合 V を K 個に $V_1, ..., V_K$ 分割する すべての分割法に対し，その各々の分割法で得られるグラフの分割が「正しい K 彩色」になっているかを調べればよい．分割のやり方は n^K 通りある．したがって，計算時間は

$$\begin{aligned}(\text{1 つの分割の正しさをチェックする時間}) \times \left(\sum_{K=1}^{n} n^K\right) &= O(n^c \times n \cdot n^n) \\ &= O^*(n^n) = O^*(2^{n \log n}).\end{aligned}$$

となる.

この指数計算時間を改善するために，表を利用したアルゴリズム[1]を考える．正確には，一度計算したことを表に蓄えておき，再計算を防ぐアルゴリズムである.

表利用アルゴリズムを設計する際には，まずはアルゴリズムを再帰的に定義すると考えやすいことが多い．ここでも次のような関数 $\mathrm{OptColor}(X)$ の計算を再帰的に考えてみよう.

$$\mathrm{OptColor}(X) = X\ (\subseteq V) \text{ だけからなる } G \text{ の部分グラフ } G[X] \text{ の最小彩色数.}$$

[1]アルゴリズムの教科書では「動的計画法」と呼ばれている場合が多い．ただ，動的計画法の場合には「計算の順序を賢く考える」という意味も含まれているように思うのだが..

これは次のように再帰的に計算できる.

```
procedure OptC(X) {
  if X = ∅ then return 0;
  min = ∞;
  for each maximal independent set I of G[X] do {
    tmp = 1 + OptC(X−I);
    if tmp < min then min = tmp;
  }
}
```

まずは，この計算が正しいことを示しておこう.

補題 1.4. 上記の手続き OptC(X) で OptColor(X) が正しく計算できる.
補注. maximal independent set（極大独立点集合）とは，その点に 1 点でも付け加えると独立点集合とならない頂点集合のこと.

次にアルゴリズムの効率について考える．この再帰型のアルゴリズムをそのまま実行すると，同じ X に対して OptC(X) を何度も計算することになる．その非効率性を避けるためには，配列 OC を用意して，一度計算した X に対しては，その値 OptC(X) を OC に格納し，再計算を防ぐようにすればよい.（具体的には X を n ビットの二進表記で表わし，それを数とみなして配列の添え字にする.）

この工夫を使い，さらに極大独立点集合を生成する際に，先週説明した MIS#1 を用いるたものを OptC#1 と呼ぶ．これに関しては次のような効率を示すことができる.

定理 1.5. OptC#1(V) の計算時間は，$O^*((1+3^{1/3})^n) = O^*((2.4423\cdots)^n)$ である.